Baseline Building Power Estimation

Sumedh Bhattacharya

Acknowledgement

I would like to thank Professor Claire Tomlin for introducing me to power systems and for her invaluable teachings and experimental advice. I also express my gratitude to Qie Hu, Datong Zhou, Ted Xiao, and the entire Hybrid Systems Laboratory at UC Berkeley for their support and guidance throughout this power systems project.

Baseline Building Power Estimation

by

Sumedh Bhattacharya

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Spring 2017

The dissertation of Sumedh Bhattacharya, titled Baseline Building Power Estimation, is approved:

Chair ______________________ Date ____________

______________________ Date ____________

University of California, Berkeley

Baseline Building Power Estimation

Abstract

Baseline Building Power Estimation

by

Sumedh Bhattacharya

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Claire Tomlin, Chair

This dissertation is two-fold: the first, and the primary focus of this project, is a study of different available techniques to perform forecasting of the power usage by the HVAC system in Sutardja Dai Hall, while the second consists of a guide to using these predictions for understanding the effects of applying energy efficient control to the building. We begin with an extensive study of different available forecasting techniques including Gaussian Mixture Models, K-Nearest Neighbors, and Recurrent Neural Networks, and a comparison of their relative accuracy in forecasting the HVAC system's power consumption. We also propose a new Hybrid Model which takes into account both time series forecasting results from past building data and weather based predictions from publicly available weather forecasting data. A high degree of accuracy is necessary as these predictions will be used to estimate the building's consumption in the absence of control. We then apply a data-driven energy efficient control model to the building to experimentally calculate savings, which had only been measured using simulations before. As such, a low error margin is required to avoid bias when calculating the deviation caused by the applied control. To this end, we see that a hybrid model using both weather and time series data achieves the highest accuracy due to its ability to model both daily and seasonal trends. Subsequently, we use these predictions to showcase an example of how to apply the results of these predictive techniques.

Contents

Contents **i**

List of Figures **iii**

List of Tables **iv**

1 Introduction **1**
- 1.1 Project Motivation 1
- 1.2 Previous Work 1
- 1.3 Contributions & Organization 2

2 Data **5**
- 2.1 BRITE Data Preprocessing 5
- 2.2 Forecast Data Preprocessing 6
- 2.3 Loss Metric Selection 7
- 2.4 Training and Testing 7

3 Pure Weather and Time Series Models **8**
- 3.1 Weather Matching Approach 8
- 3.2 Expectation Maximization Approach 10

4 Mixed Weather and Time Series Models **17**
- 4.1 Iterated Time Series Forecasting Using K-Nearest Neighbors 17
- 4.2 Multi-Input Multi-Output (MIMO) Time Series 20

5 Hybrid Model **25**
- 5.1 Weather Deviation Model 25
- 5.2 Model Combination 26
- 5.3 Performance Comparison of Hybrid vs. Individual Models 28

6 Prediction Results **30**

7 Applying Predictions to Calculate Control Gains **31**

7.1 Model Overview 31
7.2 Building Application 32
7.3 Experiment Results 33

8 Conclusion **36**
8.1 Future Work 37

Bibliography **38**

List of Figures

2.1 BRITE Breakdown 5
2.2 Time Series Data for SDH Power Consumption 6

3.1 Raw Power Readings for Best Matching Days in 2015-08-10 to 2015-09-10 9
3.2 EM Prediction with 10 Components 14
3.3 EM Prediction with 40 Components 14
3.4 Dirichlet Prior Coefficient vs. MAE 15
3.5 EM Prediction with 80 Components and Dirichlet Prior of 10^3 16

4.1 K-NN Neighbor Prediction w/ No Weather Data 19
4.2 K-NN Neighbor Prediction w/ Weather Data 19
4.3 MIMO Prediction w/ No Weather Data 23
4.4 MIMO Prediction w/ Weather Data 24

5.1 One Week Comparison of Model Predictions 29

7.1 Comparison of Baseline Consumption and Observations Under Efficient Control 34

List of Tables

3.1 Change in Accuracy by Number of Components 10
6.1 Model Prediction Result Comparison 30

Acknowledgments

I would like to thank Professor Claire Tomlin for introducing me to power systems and for her invaluable teachings and experimental advice. I also express my gratitude to Qie Hu, Datong Zhou, Ted Xiao, and the entire Hybrid Systems Laboratory at UC Berkeley for their support and guidance throughout this power systems project.

Chapter 1

Introduction

1.1 Project Motivation

Energy usage control for commercial buildings can reduce peak power consumption and stabilize overall power grid consumption by intelligently applying inputs to individual buildings. However, in order to apply intelligent control and measure its effects, an accurate tool is needed for calculating the power consumption during that same time window in the absence of control. This value is difficult to predict to an hourly degree of accuracy without requiring building specific equipment.

Specifically, in this dissertation, we study HVAC (Heating, Ventilation, and Air Conditioning) systems in these commercial buildings as they are a major source of power consumption, and their power consumption patterns follow a daily and seasonal trend. In order to understand how different power saving techniques compare to one another and calculate their overall savings, we need a mechanism to baseline the building's consumption, i.e. understand what the electricity usage of the building would have been under normal usage. This allows us to calculate the deviations caused by the application of different types of control. Ideally, the model for calculating this would not require specialized hardware to allow it to be applied to the maximum number of buildings possible. As commercial buildings account for more than 35% of the electricity consumption in the U.S., of which 39% can be attributed to the activity of HVAC systems, an application of this baselining technique would allow us to predict approximately 14% of the energy drawn on a power grid without using any additional hardware [13].

1.2 Previous Work

Support vector machines (SVMs) have been widely used in the past for load forecasting. Dong et al. [BCE05] made an early attempt in using SVMs in building energy consumption prediction. Their model used weather data and monthly utility bills as input to predict annual building energy consumption in tropical regions, and reported a percentage error within 4%. An approach using multi-agent systems at USC [Zha+15] reported a percentage error of 2.8%. We aim to reach these levels of accuracy for our prediction of the HVAC power consumption by the main supply fans in SDH but our model does not currently support the entire HVAC system and has not been tested on multiple buildings.

Using the Sutardja Dai Hall building as an example of a commercial structure equipped with a variable air volume (VAV) HVAC system, existing models from Aswani et al. [A A+12a; A A+12b] use measurements from the heating coils and pressure measurements from the air ducts to estimate the thermal baseline for the building.

$$\begin{aligned} x'(t) &= A_t(x(t)) + B_t q(x(t), u(t), v(t)) \\ y(t) &= Cx(t) \\ q(x(t), u(t), v(t)) &= q_{EW}(x(t), v(t)) + q_{win}(x(t), v(t)) \\ &\quad + q_{HVAC}(x(t), u(t), v(t)) + q_{IG}(t) \end{aligned} \tag{1.1}$$

where the state vector x is the predicted temperatures of different regions in the building. q_{EW} represents convective heat transfer and solar gains to the exterior walls, q_{win} represents convective and solar radiation gains through the windows, q_{HVAC} is a negative heat gain due to the HVAC system and q_{IG} represents internal gains due to occupancy, equipment and other uncertainties. y is then a vector containing the averaged results for the predicted temperatures in these different regions. This allows us to create a highly accurate thermal model to use as the baseline for the thermal temperatures in the room while applying control. We can then calculate the deviations from the expected baseline values to guarantee that they stay within predetermined comfort levels.

Solving this control problem is limited to only 30% of the commercial buildings that have the variable air volume (VAV) HVAC system. However, it allows us to calculate not just the deviation due to control from the expected temperature comfort zones in a building but also the difference in consumption caused by it.

1.3 Contributions & Organization

We similarly attempt to baseline power consumption to calculate the expected power consumption for the next day (24 hours) to a high degree of accuracy but without requiring specialized equipment by using only historical power readings, which are available for any BACnet connected building, and choosing to use regional weather data. This greatly increases the range of buildings for which we can understand the effects of applied control as the only data-sets necessary for applying these models are available in most commercial buildings and from publicly available repositories such as forecast.io. Furthermore, it allows for a broader spectrum of building management groups to not only predict their future energy consumption but also to prepare control algorithms, calculate their savings in terms of power usage and dollars, and compare the effectiveness of different modes of control. We augment the loss of accuracy caused by using region-specific instead of building-specific data by understanding the evolution of the building's historical power usage using time series forecasting.

In this dissertation, we initially compare the performance of three types of models: the first using Lasso regression using only weather data in Chapter 3.1, the second utilizing an EM approach using time series forecasting that reduces error immensely in Chapter 3.2, and the third using a combination of both time series forecasting (TSF) on the power readings and the weather data using K-Nearest Neighbor and Recurrent Neural Networks to gain an even finer level of accuracy in Chapter 4. We also compare their accuracy to our proposed Hybrid Model that trains two models using purely TSF and weather data respectively before intelligently weighing their results together for our final prediction in Chapter 5. A discussion of how we collected the necessary data for these models can be found in Chapter 2 and a comparison of their accuracy is recorded in Chapter 6. Finally, Chapter 7 consists of an example of how to use these predictions to understand the consumption savings from the application of intelligent control to the building.

Our first attempt fits a correlation between publicly available weather data and building power consumption to understand how changes in weather affect user activity of the HVAC system. This was able to model the general trend of the power series data but still had too high of an error to be used as a baseline value to apply control in future experiments based on. The power readings we obtain from Sutardja Dai Hall's supply fans are in a scale of 0-100, where the value represents the percentage of maximum power that is being used. For this, our goal was to be able to predict the power values to an accuracy of at least below 2 mean absolute error.

To reduce our error to these levels, we then apply the Expectation Maximization algorithm to explore the conditional dependence between time steps without the propagation of errors seen in iterative prediction. Following this, we propose a generalization of thermodynamic linear models and time series forecasting to estimate the conditional dependence on prior power use observations specific to a building.

However, our attempts to augment the feature sets of these existing models with our weather data leads to no further increases in accuracy. To overcome this, we also propose a Hybrid Model which intelligently weighs the efficacy of both a time series forecasting model and a weather based model for predicting power at different times of the day. By combining these models, we are able to not only use the current power readings to understand how the building has been functioning in the last specified window of time but also how any changes in weather may have scaled the results from what has been historical precedent. As we shall see from the results, this proves to be the most accurate and is ultimately what is used as our benchmark power consumption for the building while applying intelligent control on it to calculate the reduction in usage. When attempting to see if there is a statistical difference between the old building behavior and the one under control, we want as little error in our predictions as possible so that the deviations we witness can be attributed to the effects of control and not our model. Ultimately, the 12.6% increase in accuracy we see from the time series forecasting model to the Hybrid Model allows us to reduce the bias when using these results to compare to the readings from the building when under control. Therefore, future research can create more energy efficient control models by being able to effectively understand improvements in energy consumption that are not simply due to day-to-day variations in weather and building behavior but specifically due to the effects of the control inputs themselves.

Chapter 2

Data

2.1 BRITE Data Preprocessing

Our data consists of historic data collected from the Berkeley Retrofitted and Inexpensive HVAC Testbed for Energy Efficiency (BRITE) at Sutardja Dai Hall.

BACnet is a communication protocol that is used for building automation and control networks. Specifically, it allows for the remote control of the building's HVAC systems and lights, and gives other electronics the ability to setup building automation to communicate information between one building and another regardless of the specific service the building performs.

Using BACnet, we were able to retrieve the building's power consumption in its two main supply fans, known as AHUs (air handling units) from 01-01-2014 to 03-15-2016 in minute intervals, from the sMAP database identified in Figure 2.1. We then average these values by hour to match our weather forecast data. These readings were then preprocessed to have

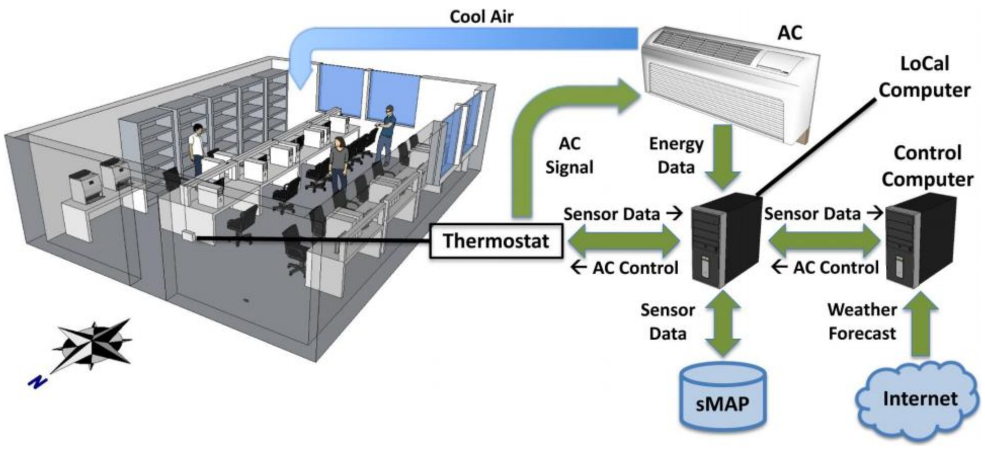


Figure 2.1: BRITE Breakdown

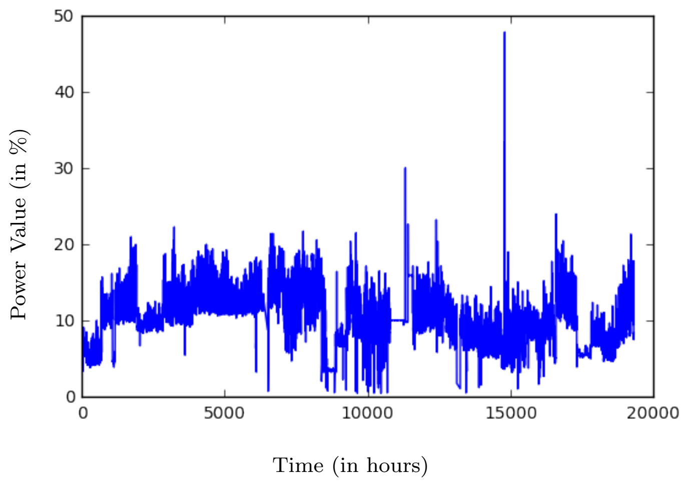


Figure 2.2: Time Series Data for SDH Power Consumption

zero mean and unit variance before being entered into our time series models. To adjust for times where the fans were manually shut off (which is a rare event that should not happen during normal functioning), we see negative readings. To compensate for this, we use linear regression to replace the values between two different valid, positive readings. These values are all then stored in a dictionary keyed, in descending hierarchical order, by day of the week, date, and hourly time.

An example view of our data is show in Figure 2.2.

2.2 Forecast Data Preprocessing

Our weather forecast data is retrieved from forecast.io's Dark Sky API, which offers both hourly and daily weather measurements for a given latitude and longitude. From their hourly readings, we obtain the following features:

- precipitation intensity, probability and type
- temperature
- apparent temperature
- humidity
- wind speed
- cloud cover

We also add to this the number of the week to account for seasonality and a binary indicator of if it is a weekend or not. We obtain this weather data for each hour of recorded power readings. Each feature in this $(19320, 10)$ array, with 19320 hourly readings (hours with missing features are dropped), is then preprocessed to have 0 mean and unit variance.

There are a few hours, usually in contiguous segments, that do not hold all of the desired features due to technical issues during that time or instrument malfunction. To account for this, all values are stored in a dictionary keyed by time. If there are any times that are missing, we also skip the appropriate key for that time in our time series data. However, this would normally cause a discontinuity within the temporal relation of a day's power readings. To ensure consistency across all of the days we are predicting on, we choose to skip that day's values entirely to ensure that we only learn on days that we have full weather data for.

Ultimately, we are left 752 valid days with complete data from both weather forecasting and building power readings.

2.3 Loss Metric Selection

For picking out our metric to measure the accuracy of our model, we looked at Zhang's study [Zha+15] that found that proposed metrics for solar power forecasting fell largely into statistical metrics for different temporal and geographical scales, uncertainty quantification metrics, ramp characterization metrics, and economic metrics. From the first category, mean absolute error was found suitable for evaluating uniform forecast error across the span of a day, and this is the metric we used to measure the accuracy of our models.

2.4 Training and Testing

For each of the models discussed in this dissertation, we use a randomized selection of 600 days and their respective hourly data points for training them and leave the remaining 152 days worth of points out for testing their accuracy while being careful not to over-fit on our training data. Furthermore, we use 10-fold cross validation on the training data to tune necessary hyper-parameters before calculating their prediction accuracy on the test data set. For our EM algorithm, we were not restricted to days with weather data available and therefore it has a different separation of training and test data identified in Chapter 3.2. Since each model requires a different partitioning of data depending on window size and the inclusion of weather data, examples of sample days of predicted power readings are presented only to provide a visual understanding of the prediction accuracy whilst the provided mean absolute error across the test set provided in Chapter 6 should be used to compare accuracy between models.

Chapter 3

Pure Weather and Time Series Models

From the temporal point of view, the simplest approach to estimate baseline consumption is that of climatology. The climatology approach consists of using a constant long-term average value throughout the entire forecasting period. This average value is then used as a benchmark for the forecasting skill with minimal effort. However, since building-specific power forecasting has surpassed this greatly, we instead approximate the baseline trend using only our weather matching model. We see that the weather matching model, using no regressive time series data, is able to approximate the general seasonal trends, and we then attempt to see if this can be replicated to a higher hourly accuracy using the opposite, a model that uses only historical time series observations for the building power. In Chapter 4, we will compare K-Nearest Neighbor and Recurrent Neural Network approaches that aim to utilize both forms of data, and in Chapter 5, we will design a model that independently uses weather and time series data before intelligently weighing when one is more accurate than the other.

3.1 Weather Matching Approach

The idea behind the weather matching approach is to create a matrix A, in which each row a_p is a difference measurement for each of our weather features between the forecasted values for the day we wish to predict and the historical day, p, we have observed. Each of these differences then has a weight stored in x which is used to predict the absolute mean corresponding difference in power values between day p and the future day f we wish to predict. While training, f is simply another historical day and we calculate the correlation between the change in weather data and change in power readings. So with n days, this gives us $n * (n - 2)$ points to train on.

Specifically, the Lasso Model is structured with features

$$w_p = \begin{bmatrix} w_{p1} & w_{p2} & \dots & w_{p10} \end{bmatrix} \tag{3.1}$$

where w_{p1} is precipitation intensity, w_{p2} is precipitation probability, and so on according to Section 2.2. Similarly, for another day f, we have

$$w_f = \begin{bmatrix} w_{f1} & w_{f2} & \dots & w_{f10} \end{bmatrix} \tag{3.2}$$

Defining a_p and γ_p, we find:

$$\begin{aligned} a_p &= w_f - w_p \\ a_{p+1} &= w_f - w_{p+1} \end{aligned} \tag{3.3}$$

$$a_j = w_f - w_j \forall j \in (p, p+n) \tag{3.4}$$

$$q = n * (n-1)/2 \tag{3.5}$$

$$\begin{bmatrix} --a_p-- \\ --a_{p+1}-- \\ \vdots \\ --a_{p+q}-- \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{10} \end{bmatrix} = \begin{bmatrix} y_p \\ y_{p+1} \\ \vdots \\ y_{p+q} \end{bmatrix} \tag{3.6}$$

$$\begin{aligned} \gamma_p &= \begin{bmatrix} \gamma_{p,1} & \gamma_{p,2} & \dots & \gamma_{p,24} \end{bmatrix} \\ \gamma_f &= \begin{bmatrix} \gamma_{f,1} & \gamma_{f,2} & \dots & \gamma_{f,24} \end{bmatrix} \end{aligned} \tag{3.7}$$

$$y_p = \text{mean}(\text{abs}(\gamma_p - \gamma_f)) \tag{3.8}$$

where γ_p are historical power readings for day p and γ_f are the power readings for day f. After training to retrieve our $x_1 ... x_{10}$ weights, we pass in the weather differences between $w_{f'}$, our future date we wish to predict, and all of our past observed days .

For example, over the course of a 30 days, from 2015-08-10 to 2015-09-10, it found the two days shown in Figure 3.1 to be the most similar:

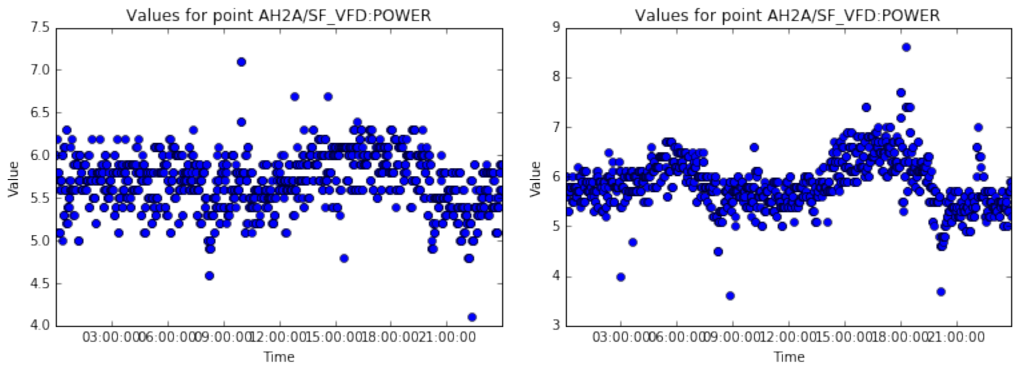


Figure 3.1: Raw Power Readings for Best Matching Days in 2015-08-10 to 2015-09-10

When making a prediction, $w_{f'}$ stores the future day's weather values and we create a new matrix A_n using differences between $w_{f'}$ and each of our historical days. Computing y' using our x weights applied on A_n correspondingly will now be the predicted mean absolute differences in power for each of our past days, and the index with the lowest magnitude marks the closest matching day to $w_{f'}$.

We then train this using linear regression with Lasso Regularization and obtain the x coefficients identified in Table 3.1 for differing number of components in PCA dimension reduction. In the table, we also see the explained variance ratio of each component in descending order as well as the mean absolute error of the reduced model's predictions when including all components above the current one:

Number of Components	Explained Variance Ratio	MAE
2	0.34663014 0.23113965	17.62
3	0.14911575	18.70
4	0.08784113	18.73
5	0.06818651	16.43
6	0.03672205	15.55
7	0.03029435	14.15
8	0.02658181	14.74
9	0.0225175	14.15
10	0.00082787	12.07

Table 3.1: Change in Accuracy by Number of Components

As we can see, dimensionality reduction does result in a loss in data but due to the increasing n^2 rate of the training size based on n days, it is helpful to note that we can reduce our dimensionality to 7 components with only a 2% cost to accuracy. Our full ten dimensional model, however, does guarantee us an accuracy of 12.07, which is the mean absolute error for each reading in comparison to the expected reading. This is still much above industry standards and we aim to lower it much further. However, this is evidence that weather data alone can model general seasonal trends in power readings, even at the hour level granularity.

3.2 Expectation Maximization Approach

Now, we switch to an expectation maximization approach to use time series forecasting to understand the conditional dependence between a historical 24-hour measurement (the recorded power readings for the day before the one we wish to predict, $d-1$) and the following 24-hours (the predicted power readings of the day d). We project the time series data points to a high dimensional regressive space where their density can then be modelled using

Gaussian Mixture Models (GMMs). This estimate of the probability density then allows us to perform short-to-medium term prediction by finding the conditional dependencies of the unknown values.

Training

First, we take a time series z consisting of n hourly power readings:

$$z_0, z_1, z_2, ..., z_{n-2}, z_{n-1}$$

We then conduct a delay embedding to create slices of window size d and arrange them in a design matrix X:

$$X = \begin{bmatrix} z_0 & z_1 & \dots & z_{d-1} \\ z_1 & z_2 & \dots & z_d \\ \vdots & \vdots & & \vdots \\ z_{n-d} & z_{n-d+1} & \dots & z_{n-1} \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-d} \end{bmatrix} \tag{3.9}$$

The density of these points can then be modelled as a Gaussian Mixture Model with the following probability density function:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{3.10}$$

where $\mathcal{N}(x|\mu_k, \Sigma_k)$ is the probability density function of the multivariate normal distribution, μ_k represents the means, Σ_k the co-variance matrices, and π_k the mixing coefficients for each component k $(0 < \pi_k < 1, \sum_{k=1}^{K} \pi_k = 1)$.

Here the number of components, which is the number of Gaussian Mixtures, can be seen as the number of different types of series of 48 hour power readings (i.e. a pairing of two adjacent days) we are learning. The means of each Gaussian Mixture represents the average values for the power reading over the course of that type of 48 hour activity. The conditional probabilities then can be intuitively viewed as a measure of how much the given time slice fitted one of the historical days we have seen. The larger number of components, the greater the different types of days we believe existed in the past.

We train this using the EM algorithm for finding a maximum-likelihood fit for log-likelihood given by

$$\log \mathcal{L}(\theta) = \log p(X|\theta) = \sum_{i=1}^{N} \log(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)) \tag{3.11}$$

where $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ are the necessary statistics that define the model.

The E-step is then to find the expected value for the log likelihood with latent variables given our design matrix and the estimation of $\theta^{(t)}$, which is the estimate of θ at timestep t given by:

$$Q(\theta|\theta^{(t)}) = E_{Z|X,\theta^{(t)}}[\log \mathcal{L}(\theta; X, Z)] \tag{3.12}$$

This then requires us to calculate the probability, r_{ik}, that sample x_i is generated by the k-th Gaussian mixture before we can maximize the log likelihood in the M-step.

$$r_{ik}^{(t)} = \frac{\pi_k^{(t)} \mathcal{N}(x_i|\mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_{j=1}^{K} \pi_{j=1}^{(t)} \mathcal{N}(x_i|\mu_j^{(t)}, \Sigma_j^{(t)})} \tag{3.13}$$

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^{(t)}) \tag{3.14}$$

We can now estimate our parameters for timestep $t+1$ with N_k being the number of samples in the k-th component:

$$\mu_k^{(t+1)} = \frac{1}{N_k} \sum_{i=1}^{N} t_{ik}^{(t)} x_i \tag{3.15}$$

$$\Sigma_k^{(t+1)} = \frac{1}{N_k} \sum_{i=1}^{N} (t_{ik}^{(t)} - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T \tag{3.16}$$

$$\pi_k^{(t+1)} = \frac{1}{N_k} \sum_{i=1}^{N} t_{ik}^{(t)} \tag{3.17}$$

The E and M step are then alternately repeated until convergence.

Predicting

Now, to predict the next future 24 hours F (unknown), we use the past 24 hour values P (known). First, we split the means and co-variances of each component into its past and future components

$$\mu_k = \begin{bmatrix} \mu_k^P \\ \mu_k^F \end{bmatrix}, \Sigma_k = \begin{bmatrix} \Sigma_k^{PP} & \Sigma_k^{PF} \\ \Sigma_k^{FP} & \Sigma_k^{FF} \end{bmatrix} \tag{3.18}$$

We can find the probability that a sample x_i^P with only past values known belongs to a component k

$$t_{ik} = \frac{\pi_k \mathcal{N}(x_i^P | \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i^P | \mu_j, \Sigma_j)} \tag{3.19}$$

Using this, we can find the conditional expectation of the future values originating from component k's F values and make a prediction $\hat{y}_i$:

$$\tilde{y}_{ik} = \mu_k^F + \Sigma_k^{FP} (\Sigma_k^{PP})^{-1} (x_i^P - \mu_k^P) \tag{3.20}$$

$$\hat{y}_i = \sum_{k=1}^{K} t_{ik} \tilde{y}_{ik} \tag{3.21}$$

Selecting Number of Components

To calculate the number of components, we picked the one that provides the least mean absolute error on the points held out for validation. To maximize our training size, we used sequential windows of hours from our total time series observations of the building instead of separating them by day. Since this is not using any weather data, we are not restricted to leaving out the days for which forecast.io was not able to provide a complete feature set. For our model selection, we used the most recent 2360 slices of size 48 hours for training our model and 591 slices of size 24 for our validation set, providing a 80-20 split.

Intuitively, increasing the number of components increases the variation in our prediction which must be balanced so as not to overfit the training data. We received our lowest error with 40 components at 2.4041669.

To visually show the difference in variation, we see an example of a fitting with 10 and 40 components in Figure 3.2 and 3.3 respectively.

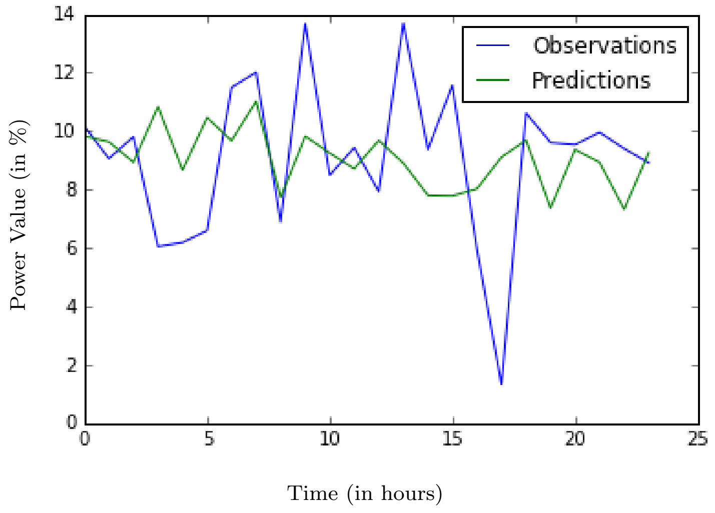


Figure 3.2: EM Prediction with 10 Components

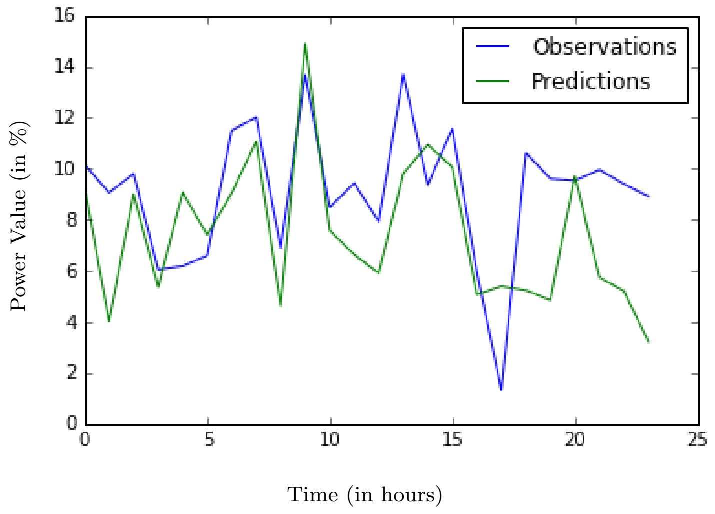


Figure 3.3: EM Prediction with 40 Components

We also wanted to understand what would happen if all of our concentrations were not weighted equally. For this, we used a Gaussian Mixture with a Dirichlet process prior fit with variational inference to add regularization in order to integrate information from prior distributions. Specifically, a low value of the concentration prior prioritizes the weight of a few components while the rest are close to zero, while a high concentration prior has more highly active components. Here, the number of specified components becomes an upper bound while the actually active components in the model are much fewer. Intuitively, this allows us to pick a set of mean days that are the most representative of usual weekly behavior and will be the most active concentrations. This has both advantages and disadvantages.

Advantages:

- Automatic component selection as a small weight concentration prior will only select a subset of the components to be active
- Less variation caused by differing number of components
- Less prone to overfitting as outlier values are regularized by lower weights

Disadvantages:

- The presence of prior weighting introduces a bias to the data and any data that does not conform to this bias will be better fitted by finite mixture models
- Requires tuning of an extra hyperparameter

Using a maximum of 80 components, we see if we can obtain a lower absolute mean error while modulating our prior in 10 steps from 10^{-2} to 10^{7}.

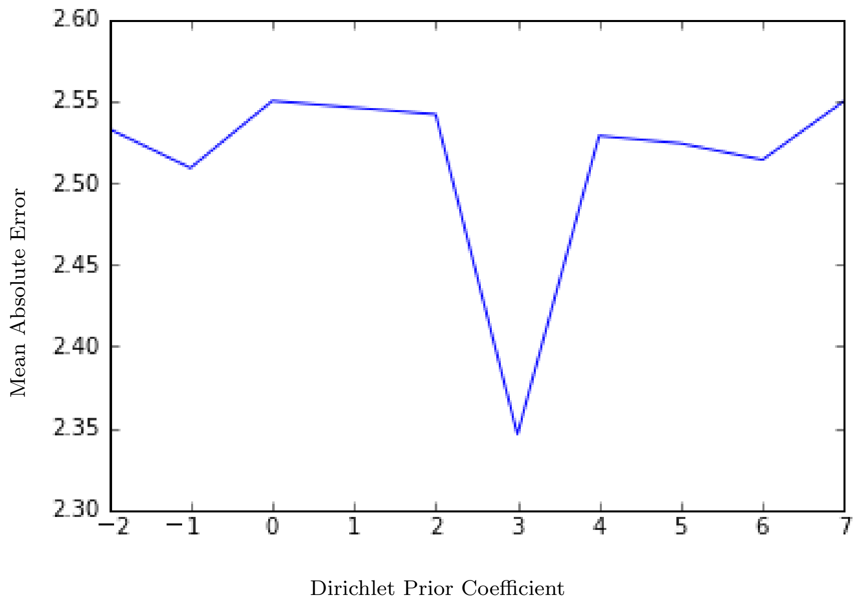


Figure 3.4: Dirichlet Prior Coefficient vs. MAE

We see the lowest error of 2.347834 at a prior value of 10^3.
The predictions for this model are shown in Figure 3.5.

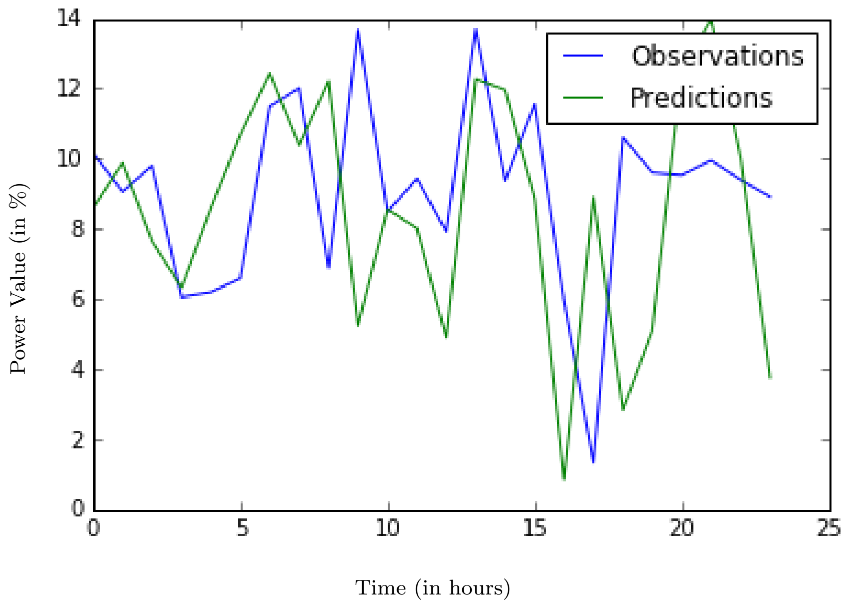


Figure 3.5: EM Prediction with 80 Components and Dirichlet Prior of 10^3

As we can see, this model is much more accurate in predicting the large variation in data due to the larger max component count without overfitting due to the presence of the prior. In fact, the graph is largely correct in estimating the peak magnitudes aside but shifts their placement by one or two hours.

Since this is most effective for stationary processes, we will later attempt to augment TS forecasting with seasonal data to account for the seasonality of power readings.

Chapter 4

Mixed Weather and Time Series Models

In order to better represent these temporal relations, we implemented more complex models, focusing on a K-Nearest Neighbors (K-NN) model and a Recurrent Neural Network (RNN). Based on the improvements of the EM Model over the baseline model, we hypothesized that a more complex model would be able to better learn the temporal relationships and dependencies inherent in the historic data. Specifically, we noticed that the EM model would underfit outliers of the model and center at the mean predictions too often.

4.1 Iterated Time Series Forecasting Using K-Nearest Neighbors

First, let's take a look at non-regressive time series forecasting. Let y_t denote the value of the time series at time point t, then we assume that

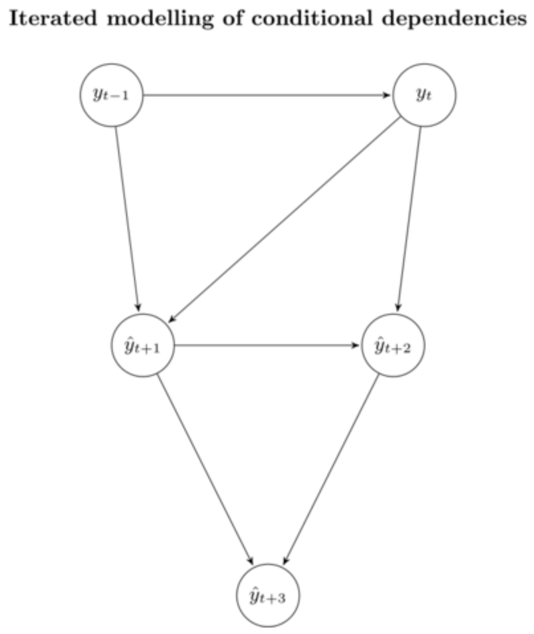
$$y_{t+1} = f(y_t, ..., y_{t-n+1}) + \epsilon_t \tag{4.1}$$

for some autoregressive order n and where ϵ_t represents some noise at time t and f is an arbitrary and unknown function. The goal is to learn this function f from the data and obtain forecasts for $t+h$, where $h \in 1, ..., H$. Hence, we are interested in predicting the next H data points, not just the H-th data point, given the history of the time series.

In iterated forecasting, we optimize a model based on predicting for a window of one step. When calculating a H-step ahead forecast, we iteratively feed the forecasts of the model back in as input for the next prediction. Essentially, what we are trying to approximate and maximize is

$$\begin{gathered} P(y_{t+1} : y_{t+H} | y_{t-n+1} : y_t) \\ y_{t+1} : y_{t+H} = [y_{t+1}, ..., y_{t+H}] \in \mathbb{R}^H \\ y_{tn+1} : y_t = [y_{t-n+1}, ..., y_t] \in \mathbb{R}^H \end{gathered} \tag{4.2}$$

where n is the autoregressive window. To visualize this, for a 3-step ahead forecast with $n = 2$, the model can be shown as:



The iterated strategy is still an unbiased estimator of $[y_{t+1} : y_{t+H}|y_{t-n+1} : y_t]$ as it preserves the stochastic dependencies of the underlying data. In terms of the bias-variance trade-off, however, the accumulation of error in the individual forecasts causes the model to have high variance. This means that we will get a low performance over longer time horizons H. To keep in parity with our EM predictions, we use 24 autoregressive terms in our historical window and iteratively predict the power readings for 24 future hours. When including the weather data, we append 10 further values containing our weather metrics for the current time to the 24 autoregressive terms to be used as features when finding matching neighbors.

The K-Nearest Neighbor (K-NN) Algorithm is a non-parametric method used for classification and regression. An unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point by Euclidean distance. We use $k = 10$ to look at the 10 most similar days in the past to predict the next day, identified by tuning this hyperparameter using 10-fold cross validation on our training set. We hope that this will overcome any averaging that occurs in similar GMM approaches we attempted in the previous section.

Sample day of power readings predicted using only iterated time series forecasting with no weather data is shown in Figure 4.1 and with weather data is shown in Figure 4.2.

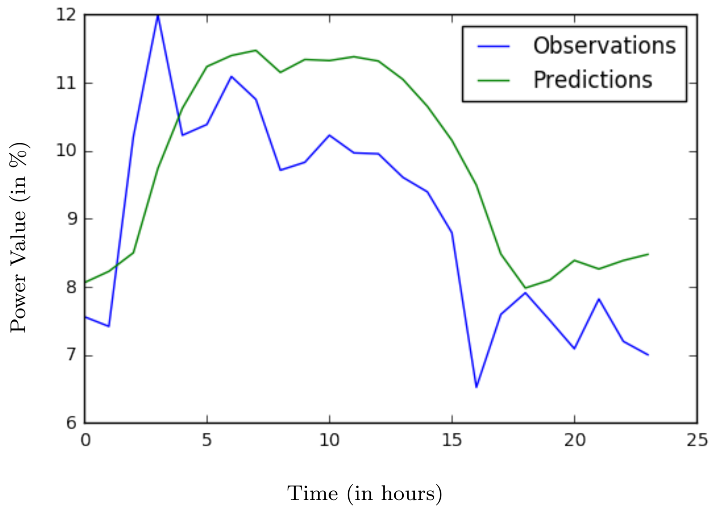


Figure 4.1: K-NN Neighbor Prediction w/ No Weather Data

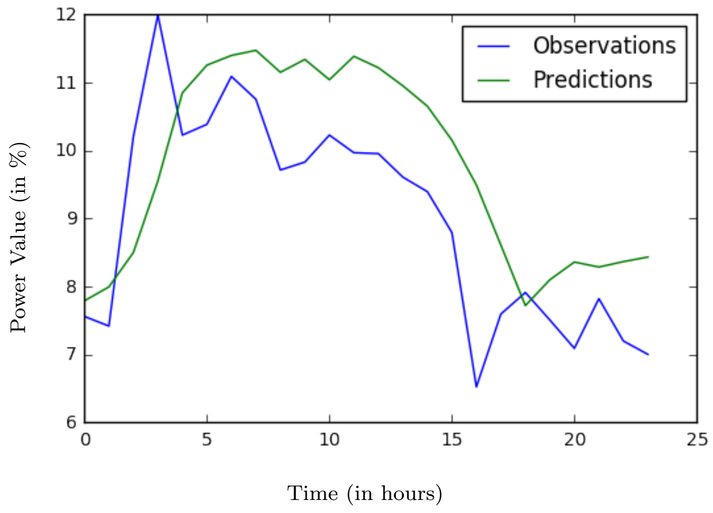


Figure 4.2: K-NN Neighbor Prediction w/ Weather Data

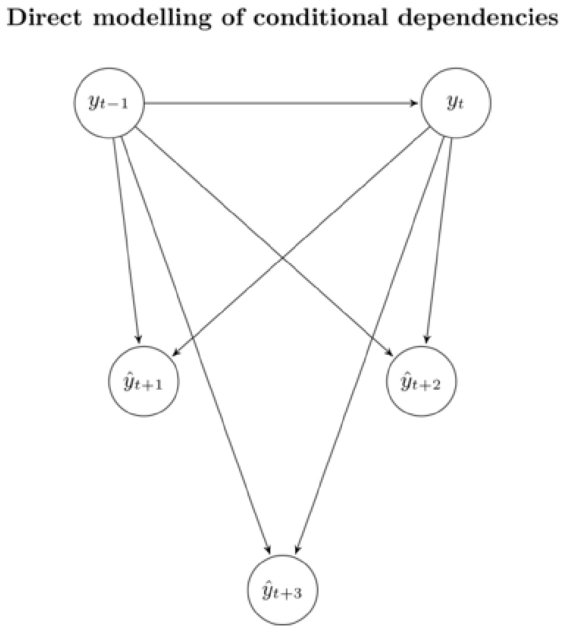
As can be seen based on the sample data alone, the inclusion of weather data does not improve our iterated time series forecasting much. Firstly, this is because the weather data for only the $t-1$-th term is used and no temporal relationship is found amongst the weather features. Secondly, our errors propagate as we step through iteration of our prediction. We can see from the graphic that the errors in our predicted values will further deviate later predicted values from being correct due to skewed conditionals. We attempt to rectify this by using a multi-output model.

4.2 Multi-Input Multi-Output (MIMO) Time Series

By using a model that trains on an H-dimensional output, where H is the size of our prediction horizon, we avoid the problem of having our errors propagate from our predictions. We then try to estimate the following function:

$$[y_{t+H}, ..., y_{t+1}] = f(y_t, ..., y_{t-n+1}) + \epsilon \tag{4.3}$$

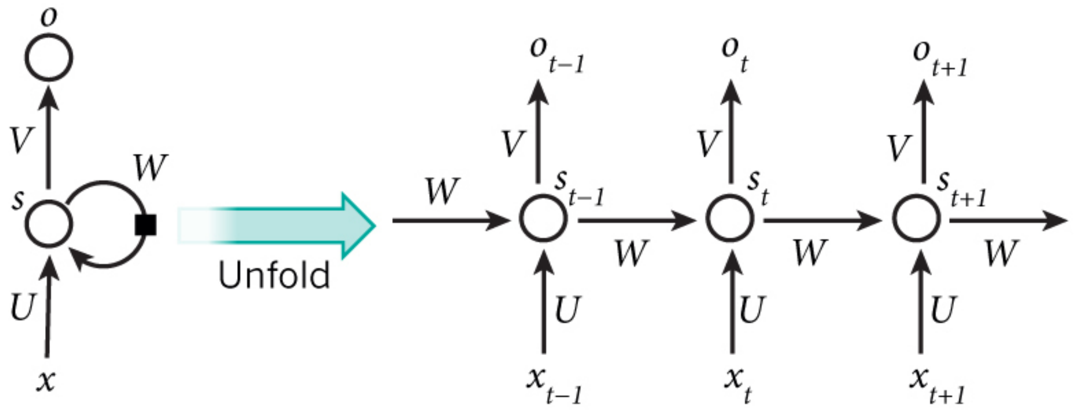
Essentially, this would be a visual example with $n = 2$ and $H = 3$:



For our multi-output model, our decision to generate our predictions using a Recurrent Neural Networks was motivated by previous RNN success in domains with temporal relationships such as natural language processing as well as short-term forecasting of the generating power of photo-voltaic cells [YSF07]. Whereas the latter was used to make predictions for up to 3 hours ahead, we aim to be able to predict a full day (24 hours) with the usage of Long Short Term Memory hidden layers.

Recurrent Neural Networks

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. The overall structure of the RNN is shown:



Here x_t is the the input to the network at time step t. For a historical window of n hours, each input to this model would be a n dimensional vector, i.e. starting at $t = 0$ we have $x = [x_0, ..., x_{n-1}]$. U, V, and W are the weights stored for each connection between the nodes of the neural network. These are derived from backpropagating through each layer of the neural network to find optimal weights that minimize the loss function, which compares the output of the final layer of the RNN and the true value of the next timestep, i.e. $o_0 = x'_n$, the model's prediction of x_n, the observed value at timestep n. Using absolute mean error and predicting H future values on each step, the loss function we minimize then is

$$L = \frac{1}{H} \sum_{i=0}^{H-1} |o_i - x_{i+n}| \tag{4.4}$$

s_t is the hidden state of the network at time step t. This is the memory of the network and is responsible for calculating the layer's output based on a combination of the previous hidden state and the current timestep's input:

$$s_t = f(Ux_t + Ws_{t-1}) \tag{4.5}$$

Here f is the non-linear activation function of this layer that allow us to predict non-linear outputs from a combination of linear layers. Without an activation function such as tanh, sigmoid, or RelU, a series of linear layers would be reduced to just one linear layer and would thus be unable to understand any non-linear relationship between historical and future values. In a regular feed-forward network, there would be no weights W as there are no connections between nodes of the same layer of the neural network. Specifically, these weights allow us to understand the influence of previous time steps stored in s_t on the current prediction. In practice, however, these typically cannot capture information beyond a small historical window due to the vanishing gradient problem. Most traditional activation functions, such as sigmoid and tanh, have gradients that fall in the range of (-1,1). However, since backpropagation computes these gradients by applying the chain rule across the hidden layers, these gradients decrease exponentially as we increase the number of layers. In recurrent neural networks, this problem also shows up when increasing the historical window due to the interlayer connections. In turn, the layers train very slowly and new inputs have a lesser impact on optimizing the weights connecting the layers.

Long Short Term Memory

To augment this, we use Long Short Term Memory (LSTM) layers instead which use a series of gates, implemented using element-wise multiplication with sigmoid functions, that learn what data should be allowed to enter and leave these hidden nodes during the iterative process of making guesses, backpropagating error, and adjusting weights via gradient descent. We can define an LSTM cell with the following equations with g as the sigmoid function and o_t as our hidden layer output at timestep t:

Input gate:

$$i_t = g(W_{xi}x_t + W_{oi}o_{t-1} + b_i) \tag{4.6}$$

Forget gate:

$$f_t = g(W_{xf}x_t + W_{of}o_{t-1} + b_f) \tag{4.7}$$

Output gate:

$$q_t = g(W_{xq}x_t + W_{oq}o_{t-1} + b_q) \tag{4.8}$$

First we transform the input using a weighted combination of the input at the current time step and the output of the last:

$$s\_in_t = tanh(W_{xs}x_t + W_{os}o_{t-1} + b_{s\_in}) \tag{4.9}$$

With this, we can now update our current timestep's hidden step with f_t weighing the value of past hidden states and i_t weighing the value of the current timestep's input:

$$s_t = f_t \cdot s_{t-1} + i_t \cdot s\_in_t \tag{4.10}$$

Finally, we can calculate the output of our hidden layer by passing the hidden-state through a final non-linear transformation and its subsequent results through our output gate:

$$o_t = q_t \cdot tanh(s_t) \tag{4.11}$$

Because of this gating mechanism, these hidden cells can retain information for longer periods of time during prediction and protect their gradients from harmful changes during training. By enforcing constant error flow through, LSTMs can learn to bridge minimal time lags in excess of 1000 discrete time steps [HS97].

RNN Models for Sutardja Dai Hall

For both of our models, we use two LSTM layers, both of size l. The first layer accepts as input a m-by-n vector where m is the number of features we are looking at and n is our historical window. For forecasting without weather data, m is 1, whereas with weather data, m is 11 as it includes all 10 features of the weather data along with our power observations from SDH. This allows us to track the influence of historical weather patterns along with consumption when forecasting power readings. It then outputs l-by-m sequences, effectively projecting our multi-feature data into a higher dimensional space on each feature. These are then condensed by the second LSTM layer into a 1-by-l vector to reduce our predictions to one feature, the power consumption. Finally, this is passed through a feed-forward layer with linear activation to reduce our dimensionality to the desired forecasting window for an output of a 1-by-H vector containing the next H predicted power readings. l, the LSTM layer size, and n, the historical window to use as input, are both hyperparameters tuned using cross-validation. We also use a dropoff rate of .33 to deactivate 1/3 of the hidden states at random on each epoch to guard against overfitting. Below, we show sample results of the models both not using weather data and using weather data with the hyperparameters that guaranteed the lowest absolute mean error.

Sample day of power readings predicted using only multi-output forecasting and no weather data:

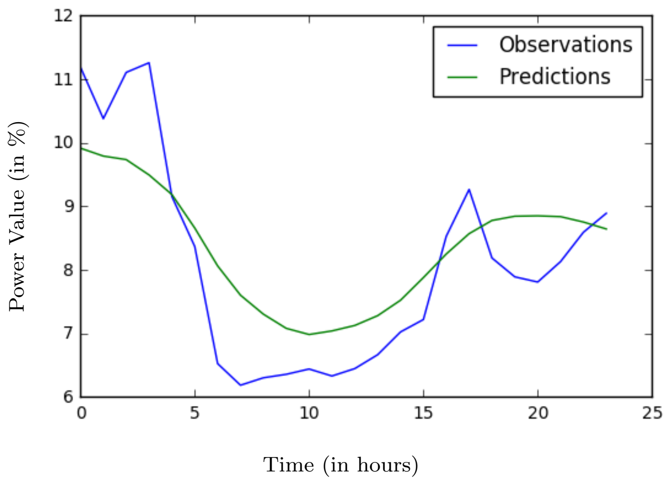


Figure 4.3: MIMO Prediction w/ No Weather Data

Absolute Mean Error: 1.10714189049 using a historical window n of size 38 and hidden layer size of 64

Sample day of power readings predicted using only multi-output forecasting and weather data:

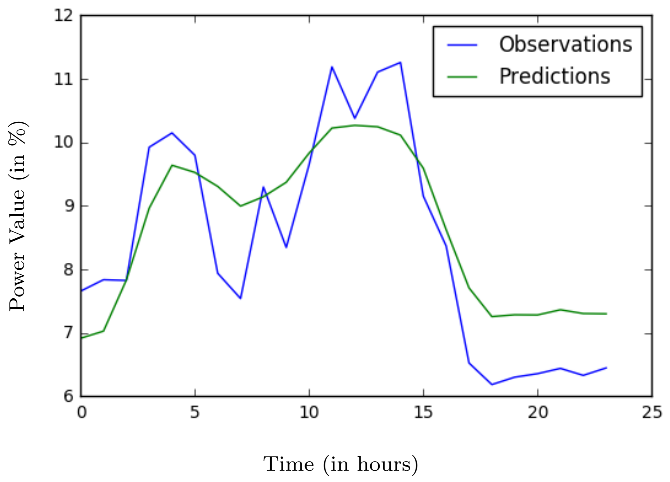


Figure 4.4: MIMO Prediction w/ Weather Data

Absolute Mean Error: 1.18180182587 using a historical window n of size 28 and a hidden layer size of 256

As we see from these results, our multi-output model using RNN's succeeds in further reducing the error of our iterative models by producing single-step forecasting results for our entire H prediction horizon and preventing the accumulation of errors. However, the inclusion of weather data actually serves to reduce our accuracy due to the lack of differentiation between the sole historical power consumption feature and the ten other weather features, leading to higher bias in favor of the weather data. To ensure that the influence of both time series power data and weather data is weighed properly, we propose the creation of a Hybrid Model where two models using each type of data independently generate predictions and then their results are combined by weighing each of their respective accuracy at each timestep of the 24 hour prediction.

Chapter 5

Hybrid Model

We create the Hybrid Model to first obtain separate individual results from our forecasting and weather based models before combining them in a manner that will maximize accuracy at each time step. Let d_f represent the day we are attempting to forecast. From the standalone time series forecasting model, we can understand how the building's current day behavior will affect its consumption for d_f, in this case the next day, whereby allowing us to see the direct relation between the power values of an adjacent autoregressive and forecast window. Our new weather model in turn looks to find the best matching historical day in terms of weather data. It then feeds these historical values into a feedforward neural net with the deviations of weather from that day to d_f as additional features to calculate how those historical values need to be modified to account for the changes in the d_f's weather from the observed past. This weather deviation based approach attempts to correct the results from our initial attempt to find the best matching day by weather in Chapter 3.1 and was applied by Senjyu et al. [Sen+02; Sen+04] for one hour ahead load forecasting and for one day ahead forecasting with online updates. We will now present a weather deviation model that produces one day ahead forecasting with only pre-experiment data. Then, we will show that we can offset the loss in accuracy by combining it with our MIMO model from Chapter 4 to create a Hybrid Model that offers the highest accuracy of all the models in this dissertation.

5.1 Weather Deviation Model

In this section, we propose a 24-hour-ahead load forecasting model that uses a similar day's data and obtains the necessary corrections to it to account for changes in weather from a feedforward neural net. In the case that there are significant changes in temperature in the current year compared to the historical weather in the building's region, we will also observe correlated changes in the building's load. In this scenario, there would be a lack of well-matched days to fetch from the past and we would be significantly limited in our ability to effectively forecast values. This model can, however, effectively correct the best matched days readings to properly reflect the building's operating conditions given the change in climate.

Senjyu et al. [Sen+04] use the minimum and maximum temperatures of each historical day compared to the current to find the best matching day. They then use online updates to calculate the load deviation from the current observed values to the best matched day's ones to understand how the differences in maximum and minimum temperature should stretch or compress the current day's predicted load curve. Since we must predict the entire 24 hours of d_f in one step, we instead choose to use the Euclidean norm of all 24 hours of the d_f's forecasted temperature to accurately find a day which has been most similar throughout the course of the prediction window. This is done by applying the algorithm for finding the best matching day defined in Section 3.1, using 24 hours of temperature readings as features instead of 10 different weather metrics. We also restrict the candidate days to only the same days of the week within 60 days before d_f and 45 days before and after d_f's date in the year prior. We label this best matching day by temperature as d_o and its power readings $y_{d_o} = [y_{d_o,0}, y_{d_o,2}, ..., y_{d_o,23}]$.

We then calculate the deviations, Δ, between d_f and d_o in terms of minimum temperature, T_{min}, maximum temperature T_{max}, average humidity U, discomfort DI, and average precipitation P. Discomfort is calculated using the following metric using $T_{d,avg}$, the average temperature for day d, and U_d, the average humidity for day d [MTB99]:

$$DI_d = 0.81T_{d,avg} + 0.01U_d(0.99T_{d,avg} - 14.3) + 46.3 \tag{5.1}$$

Deviations are calculated as follows:

$$\begin{aligned} \Delta_{T_{min}} &= T_{min,d_f}/T_{min,d_o} \\ \Delta_{T_{max}} &= T_{max,d_f}/T_{max,d_o} \\ \Delta_U &= U_{d_f}/U_{d_o} \\ \Delta_{DI} &= DI_{d_f}/DI_{d_o} \\ \Delta_P &= P_{d_f}/P_{d_o} \end{aligned} \tag{5.2}$$

The input to the feedforward neural network is then a 1-by-29 vector where

$$NN_{input} = y_{d_o} + [\Delta_{T_{min}}, \Delta_{T_{max}}, \Delta_U, \Delta_{DI}, \Delta_P] \tag{5.3}$$

and its output is a 1-by-24 vector prediction of d_f's power consumption, i.e. $NN_{output} = y'_{d_f}$.

Our feed-forward neural network model for Sutardja Dai Hall consisted of two densely connected hidden layers of dimension 512 and 128 respectively, with tanh activation functions, and a final linear output layer with dimension 24.

5.2 Model Combination

Using our individually tuned highest-accuracy models from both our MIMO forecasting and our weather deviation model, we now formulate a method using Linear Regression for combining their results in a manner that ensures we give higher weight to whichever algorithm is regularly producing more accurate results at a given time of day. Let $y'_{d_f,TSF}$ and $y'_{d_f,WD}$ be their respective 24-hour predictions for d_f. For each d_f in our training set, denoted by $d_{f,i}$ then, we separate out the results by hour such that $a_{\alpha,\beta}$ is the value for the β-th row in the matrix for the α-th hour where α ranges from 0-23.

$$\begin{bmatrix} a_{0,0} = [y_{d_{f,0},TSF,0}, y_{d_{f,0},WD,0}] \\ a_{1,0} = [y_{d_{f,0},TSF,1}, y_{d_{f,0},WD,1}] \\ \vdots \\ a_{23,0} = [y_{d_{f,0},TSF,23}, y_{d_{f,0},WD,23}] \\ a_{0,1} = [y_{d_{f,1},TSF,0}, y_{d_{f,1},WD,0}] \\ a_{1,1} = [y_{d_{f,1},TSF,1}, y_{d_{f,1},WD,1}] \\ \vdots \end{bmatrix} \tag{5.4}$$

.

Thus, we have the matrix for the the α-th hour defined as

$$A_\alpha = \begin{bmatrix} - - a_{\alpha,0} - - \\ - - a_{\alpha,1} - - \\ - - a_{\alpha,2} - - \\ \vdots \end{bmatrix} \tag{5.5}$$

Similarly, when training, we split the observed true values by hour as well and set up weights by hour, initialized to 0, for the time series forecasting and weather deviation predictions.

$$y_\alpha = [y_{d_{f,0},\alpha}, y_{d_{f,1},\alpha}, y_{d_{f,2},\alpha}, ...]^T \tag{5.6}$$

$$w_\alpha = [w_{\alpha,TSF}, w_{\alpha,WD}] \tag{5.7}$$

$$A_\alpha \cdot w_\alpha = y'_\alpha \tag{5.8}$$

We then solve 24 separate ordinary least squares linear regression problems to calculate each of the optimal weights w^*_α that obtain the most accurate predictions, y'_α, for our observed y_α values.

Now, when predicting values for a new day d_{new}, we similarly obtain $y'_{d_{new},TSF}$ and $y'_{d_{new},WD}$ from our two existing models and separate them by hour into 24 vectors of length 2. Now, our predicted power reading for the α-th hour in d_{new} is

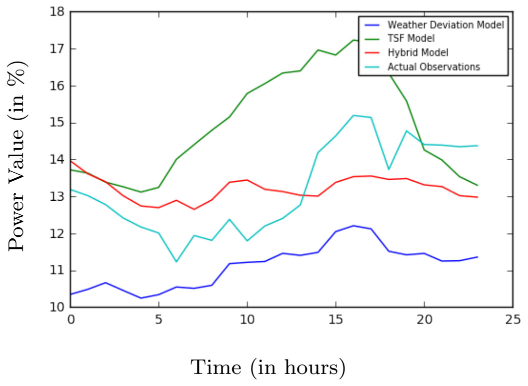
$$[y_{d_{new},TSF,\alpha}, y_{d_{new},WD,\alpha}] \cdot w^*_\alpha = y'_{d_{new},\alpha} \tag{5.9}$$

and finally our power prediction for d_{new} is

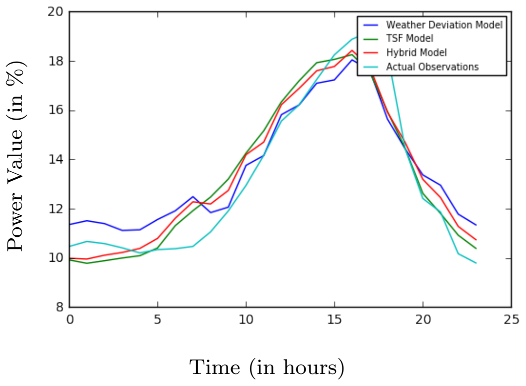
$$y'_{d_{new}} = [y'_{d_{new},0}, y'_{d_{new},1}, ..., y'_{d_{new},23}] \tag{5.10}$$

5.3 Performance Comparison of Hybrid vs. Individual Models

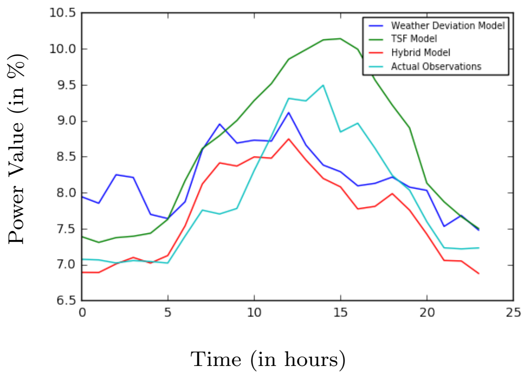
In the graphs below in Figure 5.1, we see the predicted results of each of the time series forecasting, temperature deviation, and Hybrid models plotted with the true observed power consumption values from SDH over the course of a week. The Hybrid Model is able to follow the actual observations, on average, more accurately by taking into account the over- and under-predictions of the other two models at specific time sequences and averaging out their errors. As such, we were able to achieve a MAE of less than 1, at 0.9678, across our test set, which was lower than both the weather deviation model which achieved a MAE of 1.287 and the TSF MIMO model using the RNN which achieved an MAE of 1.107. Therefore, the Hybrid Model proves to be the most accurate and versatile overall prediction model. We summarize the accuracy results of each of the models discussed thus far in the next chapter.



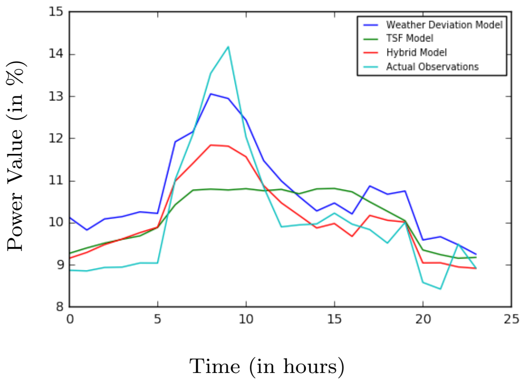
(a) Day One



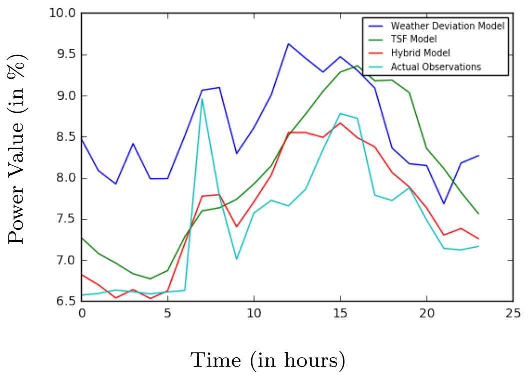
(b) Day Two



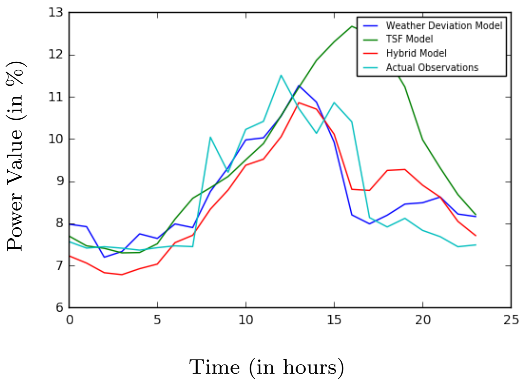
(c) Day Three



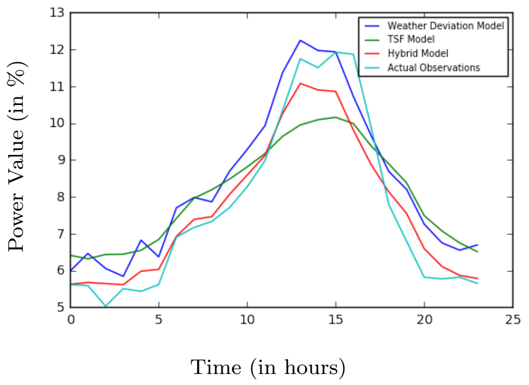
(d) Day Four



(e) Day Five



(f) Day Six



(g) Day Seven

Figure 5.1: One Week Comparison of Model Predictions

Chapter 6

Prediction Results

The following results are all measured in the absolute mean errors on 24 hour predictions on a test set of 152 slices of 24 hour historical windows for all models except for RNNs with and without weather data which had 38 hours and 28 hours respectively as tuned hyperparameters:

Model Type	**MAE**
Weather Matching w/ Lasso	12.461124
Expectation Maximization	2.404167
Expectation Maximization w/ Dirichlet Prior	2.347834
K-NN w/o Weather Data	1.256917
K-NN w/ Weather Data	1.258889
RNN w/o Weather Data	1.107142
RNN w/ Weather Data	1.181802
Weather Deviation	1.287674
Hybrid	0.96784

Table 6.1: Model Prediction Result Comparison

Chapter 7

Applying Predictions to Calculate Control Gains

In this chapter, we give an example of how we can use the predictions, specifically from our Hybrid Model, to understand the effects of applying control to the building that aims to reduce power consumption. For this, we apply model predictive control on the Data-Driven Individual Zone model devised by Datong P. Zhou, Qie Hu, and Claire J. Tomlin [ZHT16] for controlling the HVAC system in the fourth floor of Sutardja Dai Hall. This model identifies six different thermodynamic zones in the building and predicts the evolution of temperature across these zones for a specified horizon. The control then aims to minimize the airflow necessary for the 21 VAV boxes to keep the temperature in each of these zones within a comfortable bound for the defined horizon. Prior to this, the model had only been tested in simulation and had not been implemented in the actual building itself. We can now visually see the effects of the control by comparing the observed power readings from the building to our predictions from our Hybrid Model. Further work can be done in this area to apply control for the full 24 hour scope of our predictions and to use a non-parametric test, such as the Wilcoxon Signed Rank Test, to understand the statistical significance of the deviation of the observed and predicted values.

7.1 Model Overview

The Data-Driven Individual Zone model identifies the A, B, and C matrices in the following state space equation using semi-parametric regression to understand the temperature evolution of six different zones within the fourth floor of SDH:

$$\begin{aligned} x(k+1) = Ax(k) + Bu(k) + Cv(k) + q_{IG,\chi}(k) \\ \text{for } \chi \in \{F, W, S\} \end{aligned} \tag{7.1}$$

Here, x and $q_{IG,\chi}$ represent the temperature and internal thermodynamic gains of each zone respectively and u represents the airflow supplied to each zone. Depending on whether it is fall, winter, or spring, different learned internal gains will be applied to account for different levels of exogenous heating during different seasons. v is a vector containing the ambient air temperature and the HVAC system's supply air temperature. After model identification, this is used to predict the evolution of temperature across a horizon of N 15-minute timesteps for which we will apply control.

Thus, the formulation of the MPC problem to find the optimal control strategy that reduces energy consumption by minimizing the airflow necessary for the HVAC system to maintain the temperature within a comfort zone $[T_{min}, T_{max}]$ while adhering to the physical limits of the airflow in the VAV boxes themselves, defined by $[u_{min}, u_{max}]$, is:

$$\begin{aligned} \min_{u,\epsilon} \sum_{k=1}^{N} u(k)^2 + \rho\|\epsilon\|_2 \\ \text{s.t } x(0) = \bar{x}(0) \\ x(k+1) = (Eq.\ 7.1) \\ u_{min} - \epsilon \leq u(k) \leq u_{max} + \epsilon \\ T_{min} \leq x(k) \leq T_{max} \end{aligned} \tag{7.2}$$

7.2 Building Application

To apply the control to a live building, we first set up the temperature and airflow bounds in our control problem to fit those of the building itself. $\bar{x}(0)$ is the temperature of each zone at the beginning of the experiment, calculated by averaging the temperatures of rooms in each zone. As SDH uses Comfy, a web based software that takes inputs from occupants of each room to adjust the maximum and minimum temperature set-points of each room in order to cater to each person's own tastes, the bounds in each zone are different. Furthermore, the bounds within the rooms of each zone are also different as each occupant has their own personal preferences. For this, we use the averaged bounds of all rooms within each zone as the comfortable temperature range for that zone. Although Comfy defines these ranges in Fahrenheit, they were converted to Celsius to be used in the model. These values, in °C, were as follows:

$$T_{min} = [18.89, 19.44, 18.33, 19.44, 19.44, 19.44]$$
$$T_{max} = [23.89, 23.89, 23.33, 23.89, 23.89, 23.89]$$

Since this will violate the comfortable ranges for some occupants, we ran this experiment only for 6 hours, i.e. 24 15-minute timesteps, to avoid disturbing their work with our model using a receding horizon of $N = 3$ timesteps. We also heavily penalize deviation from the defined airflow input ranges by setting $\rho = 10^5$.

To run the experiment itself, we followed the steps below using a combination of a separate server running the computation on MATLAB for solving the MPC problem (Computer A) and a FitPC computer running within the building itself that listens for the inputs from MATLAB on a specified port and relays them to the building (Computer B):

1. Computer A - Fetches the current building data from sMAP to receive all temperature readings from the 4th floor of SDH.

2. Computer A - Creates the temperature trajectories for each of the six defined zones by aggregating results from the 21 VAV boxes.

3. Computer A - Loads the internal gains and learned A, B, and C matrices for the MPC problem and produces a set of N optimal inputs U^*.

4. Computer A - Takes the first optimal input, i.e. $u^* = U^*[0]$, and converts the inputs for each of the six zones to the 21 VAV boxes. Performs check on each of the VAV boxes to ensure that the inputs are bounded to the physical limits of the individual boxes even if they were beyond the bounds of the zones.

5. Computer A - Establishes connection with Computer B on a defined port and sends the VAV inputs as batched requests.

6. Computer B - Has a running listener waiting for inputs on the port Computer A is connecting to and sets the desired optimal input as both the maximum and minimum flows of the VAV box, using BACnet, to ensure that its airflow is exactly what the solution to the MPC problem dictates. It also performs another redundant check that the inputs are within the VAV boxes' physical limits to account for other modes of control that may have not performed this check.

7. Computer B - Uses BACnet to read the VAV airflows to ensure that the points were applied properly and monitor the pressure in the air ducts. If the pressure leaves the acceptable levels, i.e. it increases above 2 in. w.c. in our case, the experiment is ended.

8. Computer B - Clears all modified points to restore normal control of the building in the case of any errors except for dropped connections. If the connection is dropped, it attempts to set up a new tunnel to Computer A. If this also fails, the experiment ends and clears all points.

These steps are then repeated at each timestep. At the end of the experiment, Computer A sends a request to end the experiment and, on its arrival, Computer B proceeds to clear all modified points.

7.3 Experiment Results

During the running of the experiment from noon to 6 p.m., the maximum deviation from our defined comfort ranges for rooms that started within these ranges was .55°C with only two points, S4-02 and S4-03, going beyond bounds. The control was able to retroactively restore these rooms to within their bounds by increasing airflow to their VAV boxes, but it performed this once the temperature readings had already increased beyond the upper limit. Unfortunately, the control failed to reduce two other points, S4-04 and S4-20, that began with their temperatures already 2.78 and 1.67 °C beyond these bounds at the beginning of the experiment. These two points, however, repeatedly showcase this behavior even under normal control hinting that the limitations of the VAV airflow and increasing outside air temperatures cause these rooms to fall beyond their comfortable air ranges. The zones that these two rooms fall into are adjacent to one another and may lead to compounding heating effects due to internal gains. S4-04's zone is also adjacent to the zone that S4-02 and S4-03 fall into and S4-04's high temperature readings may also have caused the increasing temperatures seen in the latter two rooms.

Future runs of this experiment that use updated internal gains to account for Comfy usage and set up temperature bounds for each of the different rooms to account for the occupant's unique preferences can be run for longer periods of time. This will then capitalize on the MPC problem's ability to generate control that maintains temperatures within the defined range for an entire week that it has proven in simulation. Furthermore, we can also use the entire 24 hour prediction provided by our Hybrid Model along with the Wilcoxon Signed Rank Test to understand the statistical significance of the control's lowered consumption. The comparison of predicted power readings and observations from the building during the application of this experiment is shown in Figure 7.1 and the deviation in initial values for the Hybrid Model prediction and actual observations is attributed to the fact that the experiment was run from noon to 6pm instead of a full 24 hour cycle from midnight.

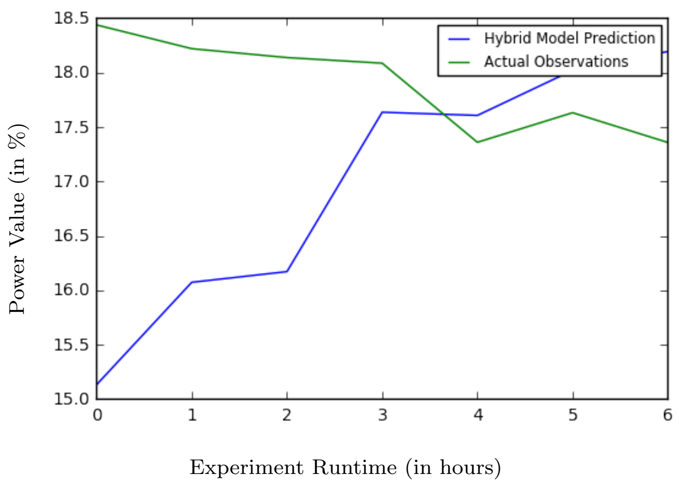


Figure 7.1: Comparison of Baseline Consumption and Observations Under Efficient Control

In this figure, we can see that after approximately 3.5 hours of the experiment running, the actual power observations from the building become lower than the Hybrid Model's predictions. While we can link this to the lasting effects of the building's normal control slowly dissipating and our optimal control reducing the power consumption of the building, this does not necessarily imply causation. To build a statistically significant link between the two, control should be applied to the building for at least the entire day and on multiple floors of the building to ensure that the effects of our control are not being diluted by consumption from the rest of the building. However, the goal of this chapter is to provide a guide, through this example, of how to apply the predictions derived from the models studied in this dissertation, specifically the new Hybrid Model.

Chapter 8

Conclusion

One of the most surprising results was the high degree of accuracy we were able to obtain for power time series prediction using only the historical power data without any weather information. While each of our progressive methods from the weather based regression approach to EM, KNN, RNN saw successively higher accuracy, the inclusion of weather, even when using temporal relationships in RNNs, saw a slight decrease or no deviation in the accuracy. This prompted us to first create two separate models for predicting power consumption using TSF and a weather deviation model to account for changes in the region's weather patterns and then combine them using linear regression to achieve our highest accuracy.

While the weather series data alone had a high error at an hourly level, we were able to reduce our error drastically using EM. However, one of the problems we faced using EM was the low variation in its predicted values that would not suitably match the peaks in the observed data. To combat this, we used a higher number of components while being cautious about overfitting. Further, we observed that using a Dirichlet Prior allowed us to predict the peaks highly accurately but still had an error above 2 due to the offsets in peak position in the range of a few hours. To combat this, we applied time series forecasting that concretely modeled this temporal relation in time series data in two manners, iteratively (using K-NN) and directly (using RNN). The latter was able to predict the power consumption in Sutardja Dai Hall by the main supply fans of its HVAC system with a lowest mean absolute error of 1.107142. Finally, we incorporated in our results from our original weather matching model by allowing for a neural net to calculate changes in the power curve based on the deviation on different weather features. When we combined this with our RNN model that used no weather data using our Hybrid Model architecture, we were able to reach our highest accuracy of 0.96784.

8.1 Future Work

We can further extend the predictive models in work to model the power consumption of all electrical devices, including HVAC, lighting, and appliances, in the BRITE Sutardja Dai Hall system and then further extend our training set to multiple buildings to see how well it learns for other commercial buildings located in various different districts, housing different types of services, and geo-spatially well separated. This will allow us to see how well each of these models adapt to the various different types of buildings that may all be present on one power grid.

Furthermore, once we have trained the model to predict the power usage patterns for multiple buildings on a power grid, we can apply intelligent control for peak shaping while having an accurate baseline to understand our deviation.

Bibliography

[HS97] Sepp Hochreiter and Jurgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation 9(8)*. 1997, pp. 1735–1780.

[MTB99] P. A. Mastorocostas, J. B. Theocharis, and A. G. Bakirtzis. "Accumulative effect of discomfort index for fuzzy short-term load forecasting". In: *IEEE Transactions on Power Systems*. Vol. 14. 1999, pp. 29–36.

[Sen+02] T. Senjyu et al. "One-Hour-Ahead Load Forecasting Using Neural Network". In: *IEEE Transactions on Power Systems*. Vol. 17. IEEE, 2002.

[Sen+04] T. Senjyu et al. "Next day load curve forecasting using hybrid correction method". In: *IEEE Proceedings - Generation, Transmission and Distribution*. Vol. 151. IEEE, 2004.

[BCE05] B. Dong, C. Cao, and E.L. Siew. "Applying Support Vector Machines to Predict Building Energy Consumption in Tropical Regions". In: *Energy and Buildings*. Vol. 37. Elsevier Ltd., 2005.

[YSF07] A. Yona, T. Senjyu, and T. Funabashi. "Application of Recurrent Neural Network to Short-Term-Ahead Generating Power Forecasting for Photovoltaic System". In: *Power Engineering Society General Meeting*. IEEE, 2007.

[A A+12a] A. Aswani et al. "Energy-Efficient Building HVAC Control Using Hybrid System LBMPC". In: *IFAC Proceedings Volumes*. Vol. 45. Elsevier Ltd., 2012, pp. 496–501.

[A A+12b] A. Aswani et al. "Identifying Models of HVAC Systems Using Semiparametric Regression". In: *American Control Conference*. ACC. 2012.

[EL13] E. Eirola and A. Lendasse. "Gaussian Mixture Models for Time Series Modelling, Forecasting, and Interpolation". In: *International Symposium on Intelligent Data Analysis*. Vol. 8207. IDA. 2013.

[Li+13] N. Li et al. "Predicting HVAC Energy Consumption in Commercial Buildings Using Multiagent Systems". In: *30th International Symposium on Automation and Robotics in Construction*. ISARC. 2013.

[13] "The Annual Energy Outlook 2013". In: *Tec. Rep. '13*. US Energy Information Administration, 2013.

[Zha+15] J. Zhang et al. “A Suite of Metrics for Assessing the Performance of Solar Power Forecasting”. In: *Solar Energy*. Vol. 111. Elsevier Ltd., 2015.

[Q H+16] Q. Hu et al. “Model Identification of Commercial Building HVAC Systems during Regular Operation - Empirical Results and Challenges”. In: *American Control Conference*. ACC. 2016.

[ZHT16] Datong Zhou, Qie Hu, and Claire J. Tomlin. “Model Comparison of a Data-Driven and a Physical Model for Simulating HVAC Systems”. In: 2016.